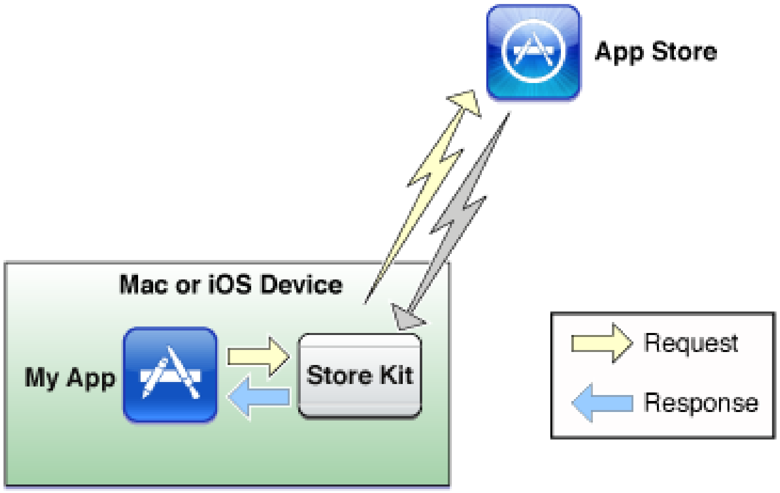
EXHIBIT 9

Overview of In-App Purchase

Store Kit communicates with the App Store on behalf of your application. Your application uses Store Kit to receive localized information from the App Store about products you want to offer in your application. Your application displays this information to users and allows them to purchase items. When a user wants to purchase an item, your app calls Store Kit to collect payment from the user. Figure 1-1 shows the basic store model.

Figure 1-1 In-App Store model



The Store Kit API is only a small part of the process of adding a store to your application. You need to decide how to track the products you plan to deliver, how your application presents a store front to the user, and how your application delivers the products users purchase from your store. The rest of this chapter explains the process of creating products and adding a store to your application.

Products

A product is any feature that you want to sell in your application's store. Products are associated with the App Store through iTunes Connect in the same way that you create new applications. There are four supported kinds of products that you may sell using In-App Purchase:

- Content includes digital books, magazines, photos, artwork, game levels, game characters, and other digital content that can be delivered within your application.
- Functionality products unlock or expand features you've already delivered in your application. For example, you could ship a game with multiple smaller games that could be purchased by the user.
- Services allow your application to charge users for one-time services, such as voice transcription. Each time the service is used is a separate purchase.
- Subscriptions provide access to content or services on an extended basis. For example, your application might offer monthly access to financial information or to an online game portal.

In-App Purchase provides a general mechanism for creating products, leaving the specifics of how your products are implemented up to you. However, there are few important guidelines to keep in mind as you design your application:

- You must deliver a digital good or service within your application. Do not use In-App Purchase to sell real-world goods and services.
- You may not offer items that represent intermediary currency because it is important that users know the specific good or service they are buying.
- Items you offer for purchase may not contain, or relate to, pornography, hate speech, defamation, or gambling (simulated gambling is acceptable).

For detailed information about what can be offered using In-App Purchase, consult your licensing agreement.

Registering Products with the App Store

Every product you wish to offer in your store must first be registered with the App Store through iTunes Connect. When you register a product, you provide a name, description, and pricing for your product, as well as other metadata used by the App Store and your application.

You identify a particular product using a unique string called a product identifier. When your application uses Store Kit to communicate with the App Store, it uses product identifiers to retrieve the configuration data you provided for the product. Later, when a customer wants to purchase a product, your application identifies the product to be purchased using its product identifier.

The App Store supports many types of products:

- Consumable products must be purchased each time the user needs that item. For example, one-time services are commonly implemented as consumable products.
- Non-consumable products are purchased only once by a particular user. Once a non-consumable product is purchased, it is provided to all devices associated with that user's iTunes account. Store Kit provides built-in support to restore non-consumable products on multiple devices.
- Auto-renewable subscriptions are delivered to all of a user's devices in the same way as non-consumable products. However, auto-renewable subscriptions differ in other ways. When you create an auto-renewable subscription in iTunes Connect, you choose the duration of the subscription. The App Store automatically renews the subscription each time its term expires. If the user chooses to not allow the subscription to be renewed, the user's access to the subscription is revoked after the subscription expires. Your application is responsible for validating whether a subscription is currently active and can also receive an updated receipt for the most recent transaction.
- Free subscriptions are a way for you to put free subscription content in Newsstand. Once a user signs up for a free subscription, the content is available on all devices associated with the user's Apple ID. Free subscriptions do not expire and can only be offered in Newsstand-enabled apps.
- Non-renewing subscriptions are a mechanism for creating products with a limited duration. Non-renewing subscriptions differ from auto-renewable subscriptions in a few key ways:
    - The term of the subscription is not declared when you create the product in iTunes Connect; your application is responsible for providing this information to the user. In most cases, you would include the term of the subscription in the description of your product.

- Non-renewing subscriptions may be purchased multiple times (like a consumable product) and are not automatically renewed by the App Store. You are responsible for implementing the renewal process inside your application. Specifically, your application must recognize when the subscription has expired and prompt the user to purchase the product again.
- You are required to deliver non-renewing subscriptions to all devices owned by the user. Non-renewing subscriptions are not automatically synchronized to all devices by Store Kit; you must implement this infrastructure yourself. For example, most subscriptions are provided by an external server; your server would need to implement a mechanism to identify users and associate subscription purchases with the user who purchased them.

Detailed information about registering products with the App Store can be found in iTunes Connect Developer Guide.

Feature Delivery

The delivery mechanism your application uses to provide products to users has significant implications on its design and implementation. There are two basic models you should expect to use to deliver products to users: the built-in model and the server model. In both models, you track the list of products offered in the store and deliver products successfully purchased by users.

Built-in Product Model

In the built-in product model, everything required to deliver products is built in to your application. This model is most often used to unlock functionality in your application. You could also use this model to deliver content provided in your application's bundle. A key advantage of this model is that your application can promptly deliver products to the customer. Most built-in products should be non-consumable.

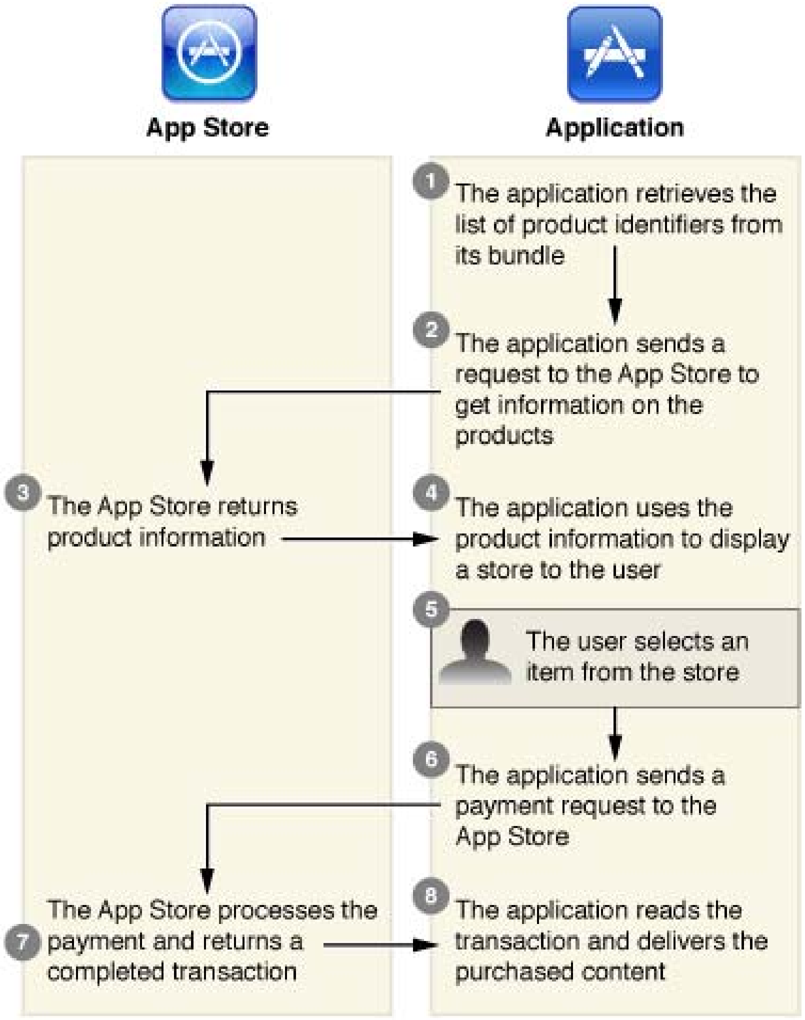
> Important: In-App Purchase does not provide the capability for your application to be patched after a successful purchase. If your product requires changes to your application's bundle, you must deliver an updated version of your application to the App Store.

To identify products, your application stores the product identifiers in your application's bundle. Apple recommends using a property list (plist) to track product identifiers for your built-in features. Content-driven applications can use this to add new content without modifying the source for your application.

After a product is successfully purchased, your application must unlock the feature and deliver it to the user. The simplest way to unlock features is by changing your application preferences. See "Implementing Application Preferences". Application preferences are backed up when users backs up their iOS-based devices. Your application may want to recommend to users that they back up their devices after making a purchase to ensure that purchases are not lost.

Figure 1-2 shows the series of actions your application takes to deliver a built-in product.

Figure 1-2 Built-in product delivery

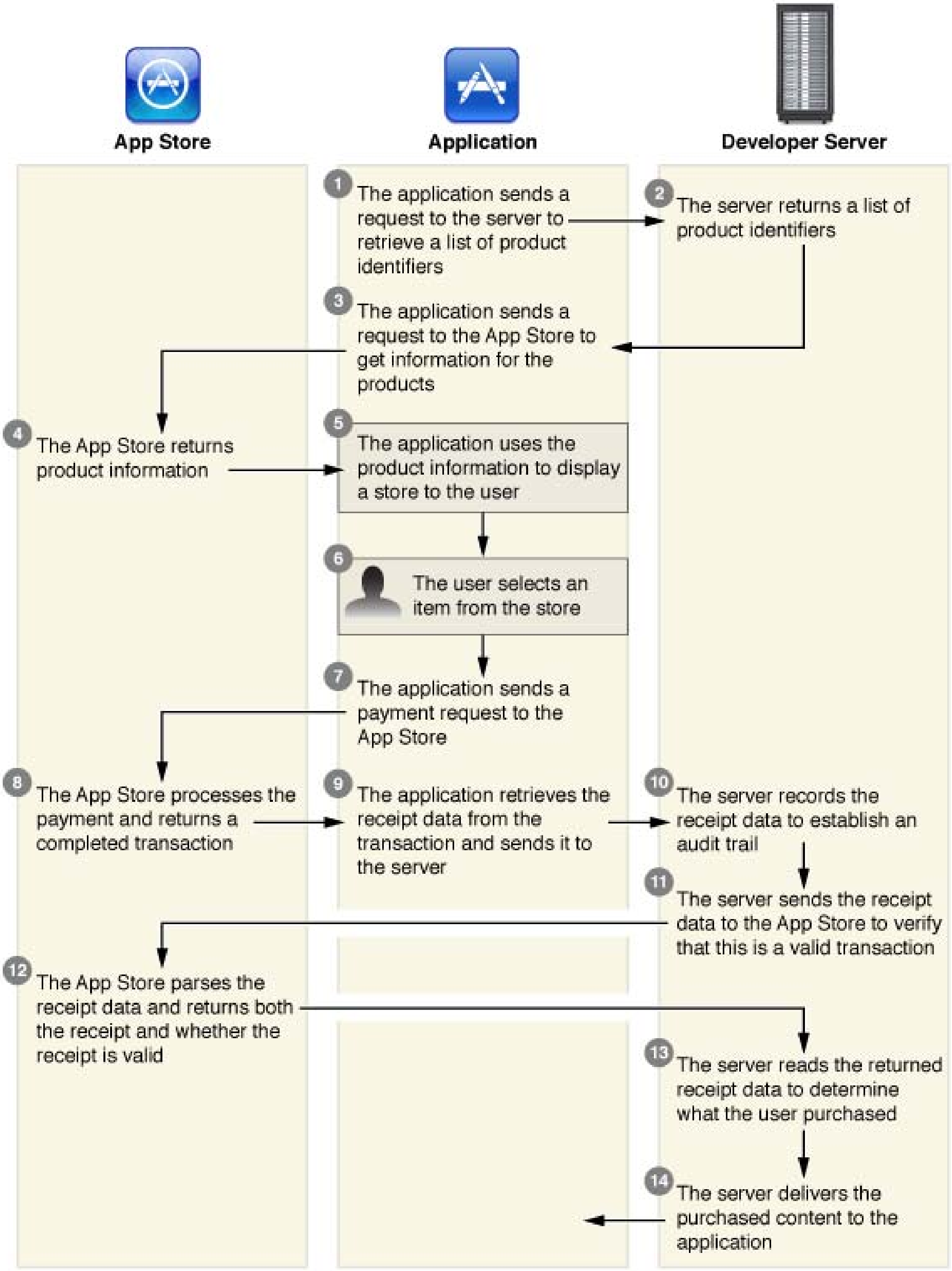

App Store
Application
1 The application retrieves the list of product identifiers from its bundle
2 The application sends a request to the App Store to get information on the products
3 The App Store returns product information
4 The application uses the product information to display a store to the user
5 The user selects an item from the store
6 The application sends a payment request to the App Store
7 The App Store processes the payment and returns a completed transaction
8 The application reads the transaction and delivers the purchased content

Server Product Model

In the server product model, you provide a separate server that delivers products to your application. Server delivery is appropriate for subscriptions, services and content, because these products can be delivered as data without altering your application bundle. For example, a game might deliver new play environments (puzzles or levels) to the application. Store Kit does not define the design of your server or its interactions with your application. You are responsible for designing all interactions between your application and your server. Further, Store Kit does not provide a mechanism to identify a particular user. Your design may require you to provide a mechanism to identify a user. If your application requires these (for example, to track which subscriptions are associated with a particular user), you need to design and implement this yourself.

Figure 1-3 expands the built-in model to show interactions with a server.

Figure 1-3 Server product delivery


App Store
Application
Developer Server
1 The application sends a request to the server to retrieve a list of product identifiers
2 The server returns a list of product identifiers
3 The application sends a request to the App Store to get information for the products
4 The App Store returns product information
5 The application uses the product information to display a store to the user
6 The user selects an item from the store
7 The application sends a payment request to the App Store
8 The App Store processes the payment and returns a completed transaction
9 The application retrieves the receipt data from the transaction and sends it to the server
10 The server records the receipt data to establish an audit trail
11 The server sends the receipt data to the App Store to verify that this is a valid transaction
12 The App Store parses the receipt data and returns both the receipt and whether the receipt is valid
13 The server reads the returned receipt data to determine what the user purchased
14 The server delivers the purchased content to the application

Apple recommends you retrieve product identifiers from your server, rather than including them in a property list. This gives you the flexibility to add new products without updating your application.

In the server model, your application retrieves the signed receipt associated with a transaction and sends it to your server. Your server can then validate the receipt and decode it to determine which content to deliver to your application. This process is covered in detail in “Verifying Store Receipts.”

The server model has additional security and reliability concerns. You should test the entire environment for security threats. Secure Coding Guide provides additional recommendations.

Although non-consumable products may be recovered using the built-in capabilities of Store Kit, non-renewing subscriptions must be restored by your server. You are responsible for recording information about non-renewing subscriptions and restoring them to users. Optionally, consumable products could also be tracked by your server. For example, if your consumable product is a service provided by your server, you may want the user to retrieve the results of that request on multiple devices.